Importing the Libraries and reaading the dataset

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.impute import KNNImputer
import re
from datetime import datetime
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import sys
import warnings
warnings.filterwarnings("ignore")
import scipy.cluster.hierarchy as hc
```

In [2]:
```
data = pd.read_csv('scaler_clustering.csv')
data.head()
```

Out[2]:

	Unnamed: 0	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year
0	0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	Other	2020.0
1	1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0	449999	FullStack Engineer	2019.0
2	2	ojzwnvwnxw vx	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	2015.0	2000000	Backend Engineer	2020.0
3	3	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	2017.0	700000	Backend Engineer	2019.0
4	4	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	2017.0	1400000	FullStack Engineer	2019.0

Dropping the unwanted columns

In [3]:
```
data.drop(columns = 'Unnamed: 0', axis= 1, inplace = True)
```

Checking Basic Metrics of the Dataset

In [4]:
```
data.describe()
```

Out[4]:

	orgyear	ctc	ctc_updated_year
count	205757.000000	2.058430e+05	205843.000000
mean	2014.882750	2.271685e+06	2019.628231
std	63.571115	1.180091e+07	1.325104
min	0.000000	2.000000e+00	2015.000000
25%	2013.000000	5.300000e+05	2019.000000
50%	2016.000000	9.500000e+05	2020.000000
75%	2018.000000	1.700000e+06	2021.000000
max	20165.000000	1.000150e+09	2021.000000

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205843 entries, 0 to 205842
Data columns (total 6 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   company_hash      205799 non-null  object 
 1   email_hash        205843 non-null  object 
 2   orgyear           205757 non-null  float64
 3   ctc               205843 non-null  int64  
 4   job_position      153279 non-null  object 
 5   ctc_updated_year  205843 non-null  float64
dtypes: float64(2), int64(1), object(3)
memory usage: 9.4+ MB
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
company_hash           44
email_hash              0
orgyear                86
ctc                     0
job_position        52564
ctc_updated_year        0
dtype: int64
```

In [7]:
```
data.size
```

Out[7]:
```
1235058
```

In [8]:
```
data.shape
```

Out[8]:
```
(205843, 6)
```

Checking Unique values for required columns

In [9]:
```
s = ['company_hash', 'email_hash', 'job_position']

for i in s:
    print(f"Top 5 values for {i}:")
    print(data[i].value_counts().head(5))
    print("-------------------------")
```

```
Top 5 values for company_hash:
company_hash
nvnv wgzohrnvzwj otqcxwto    8337
xzegojo                      5381
vbvkgz                       3481
zgn vuurxwvmrt vwwghzn       3411
wgszxkvzn                    3240
Name: count, dtype: int64
-------------------------
Top 5 values for email_hash:
email_hash
bbace3cc586400bbc65765bc6a16b77d8913836cfc98b77c05488f02f5714a4b    10
6842660273f70e9aa239026ba33bfe82275d6ab0d20124021b952b5bc3d07e6c     9
298528ce3160cc761e4dc37a07337ee2e0589df251d73645aae209b010210eee     9
3e5e49daa5527a6d5a33599b238bf9bf31e85b9efa9a94f1c88c5e15a6f31378     9
b4d5afa09bec8689017d8b29701b80d664ca37b83cb883376b2e95191320da66     8
Name: count, dtype: int64
-------------------------
Top 5 values for job_position:
job_position
Backend Engineer          43554
FullStack Engineer        24717
Other                     18071
Frontend Engineer         10417
Engineering Leadership     6870
Name: count, dtype: int64
-------------------------
```

In [10]:

```python
counts = data.groupby(['company_hash', 'job_position']).size().reset_index(name='count')
counts = counts.sort_values(by='count', ascending=False)
print(counts)
```

```
                   company_hash        job_position  count
21331  nvnv wgzohrnvzwj otqcxwto    Backend Engineer   1355
41496                      vbvkgz    Backend Engineer   1159
21358  nvnv wgzohrnvzwj otqcxwto               Other   1058
11983                      gqvwrt    Backend Engineer    903
21345  nvnv wgzohrnvzwj otqcxwto  FullStack Engineer    871
...                          ...                 ...    ...
22382   nyt nxbto ge xzaxv sqghu               Other      1
22383         nyt nxbto xzntqztn  FullStack Engineer      1
22384            nyt nxmtq sqghu  FullStack Engineer      1
22385             nyt obvqn whmt        Data Analyst      1
58985                      zzzbzb               Other      1

[58986 rows x 3 columns]
```

In [11]:
```
data['email_hash'].nunique()
```

Out[11]:
```
153443
```

In [12]:
```
filtered_data = data[data['email_hash'] == 'bbace3cc586400bbc65765bc6a16b77d8913836cfc98b77c05488f02f5714a4b']
print(filtered_data)
```

```
                    company_hash  \
24109   oxej ntwyzgrgsxto rxbxnta
45984   oxej ntwyzgrgsxto rxbxnta
72315   oxej ntwyzgrgsxto rxbxnta
102915  oxej ntwyzgrgsxto rxbxnta
117764  oxej ntwyzgrgsxto rxbxnta
121483  oxej ntwyzgrgsxto rxbxnta
124476  oxej ntwyzgrgsxto rxbxnta
144479  oxej ntwyzgrgsxto rxbxnta
152801  oxej ntwyzgrgsxto rxbxnta
159835  oxej ntwyzgrgsxto rxbxnta

                                              email_hash  orgyear     ctc  \
24109   bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  720000
45984   bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  720000
72315   bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  720000
102915  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  720000
117764  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  720000
121483  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  660000
124476  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  660000
144479  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  660000
152801  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  660000
159835  bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...    2018.0  660000

             job_position  ctc_updated_year
24109                 NaN            2020.0
45984     Support Engineer            2020.0
72315                Other            2020.0
102915  FullStack Engineer            2020.0
117764       Data Analyst            2020.0
121483              Other            2019.0
124476    Support Engineer            2019.0
144479  FullStack Engineer            2019.0
152801     Devops Engineer            2019.0
159835                NaN            2019.0
```

Preprocessing the Data

In [13]:

```python
def preprocess_string(string):
    new_string= re.sub('[^A-Za-z ]+', '', string).lower().strip()
    return new_string
```

```
mystring='\tAirtel\\\\&&**() X Labs'
preprocess_string(mystring)

data['job_position']=data.job_position.apply(lambda x: preprocess_string(str(x)))
data['company_hash']=data.company_hash.apply(lambda x: preprocess_string(str(x)))
```

Dropping the Null columns from Company hash and Job position

In [14]:
```
data=data[ ~((data['company_hash']=='') | (data['job_position']==''))]
```

In [15]:
```
data.isna().sum()
```

Out[15]:
```
company_hash         0
email_hash           0
orgyear             86
ctc                  0
job_position         0
ctc_updated_year     0
dtype: int64
```

In [16]:
```
knn_imputer = KNNImputer(n_neighbors=3)
# Select only numerical columns for KNN imputation
numerical_cols = ['orgyear']
data[numerical_cols] = knn_imputer.fit_transform(data[numerical_cols])
# Convert orgyear back to int
data['orgyear'] = data['orgyear'].astype(int)
```

In [17]:
```
data.isna().sum()
```

Out[17]:
```
company_hash        0
email_hash          0
orgyear             0
ctc                 0
job_position        0
ctc_updated_year    0
dtype: int64
```

Univaraiate Analysis

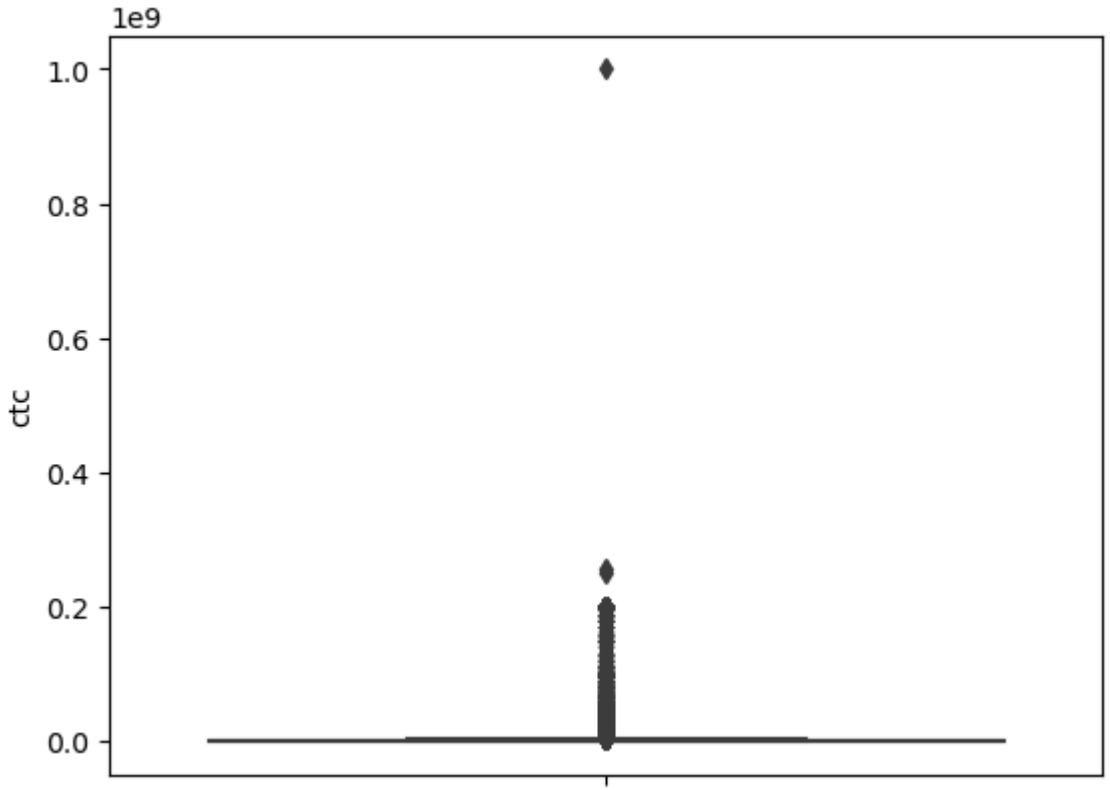
In [18]:
```
data
```

Out[18]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016	1100000	other	2020.0
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018	449999	fullstack engineer	2019.0
2	ojzwnvwnxw vx	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	2015	2000000	backend engineer	2020.0
3	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	2017	700000	backend engineer	2019.0
4	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	2017	1400000	fullstack engineer	2019.0
...	...	...	...	...	...	...
205838	vuurt xzw	70027b728c8ee901fe979533ed94ffda97be08fc23f33b...	2008	220000	nan	2019.0
205839	husqvawgb	7f7292ffad724ebbe9ca860f515245368d714c84705b42...	2017	500000	nan	2020.0
205840	vwwgrxnt	cb25cc7304e9a24facda7f5567c7922ffc48e3d5d6018c...	2021	700000	nan	2021.0
205841	zgn vuurxwvmrt	fb46a1a2752f5f652ce634f6178d0578ef6995ee59f6c8...	2019	5100000	nan	2019.0
205842	bgqsvz onvzrtj	0bcfc1d05f2e8dc4147743a1313aa70a119b41b30d4a1f...	2014	1240000	nan	2016.0

205745 rows × 6 columns

In [19]:
```
sns.boxplot(data = data,y = 'ctc')
plt.show()
```



In [20]:
```
data[data['ctc']>1000000000]
```

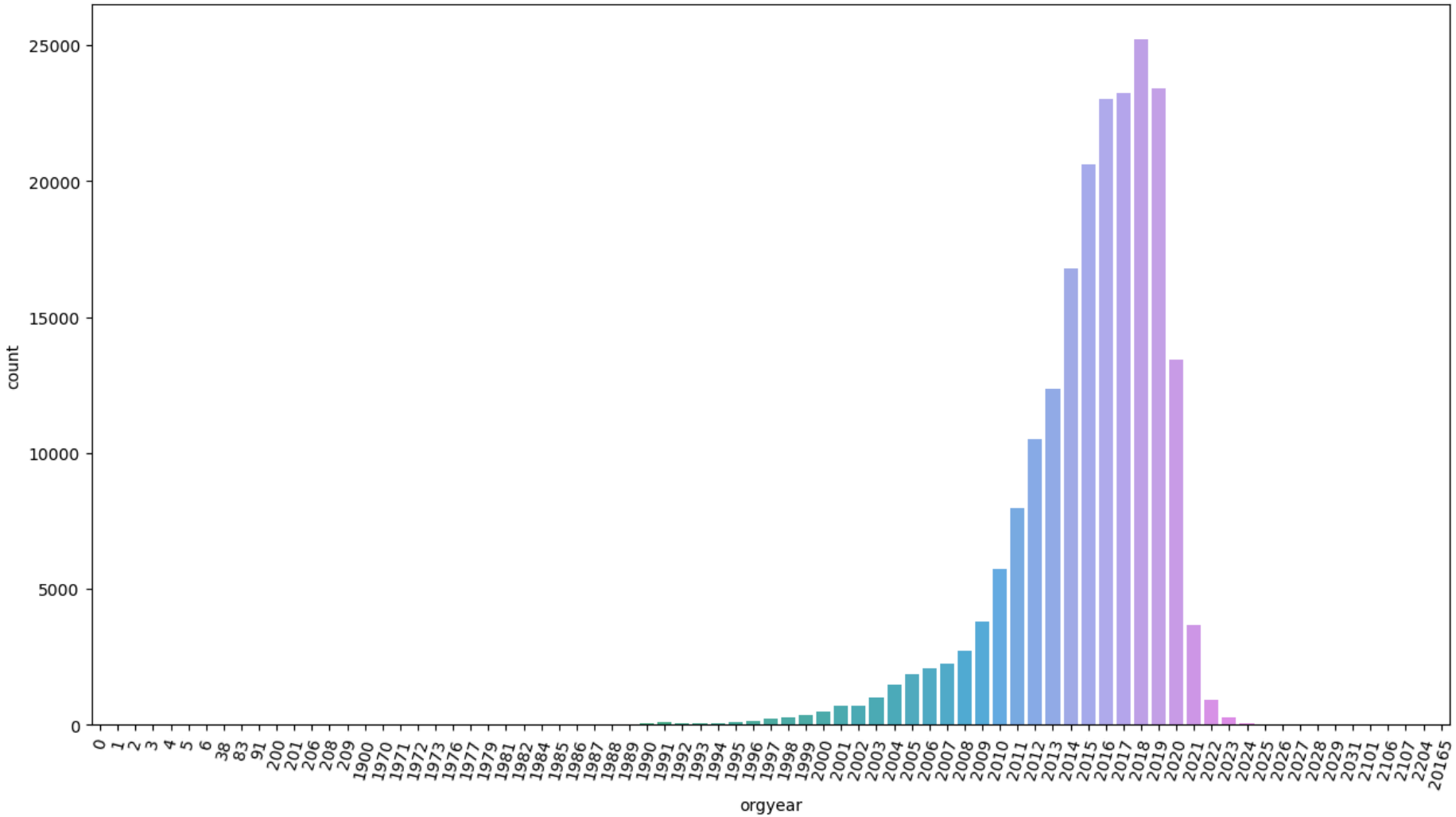
Out[20]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year
72824	whmxw rgsxwo uqxcvnt rxbxnta	29a71dd13adf6d2d497571a565bb3096cf66cb46cd1ece...	2015	1000150000	nan	2020.0

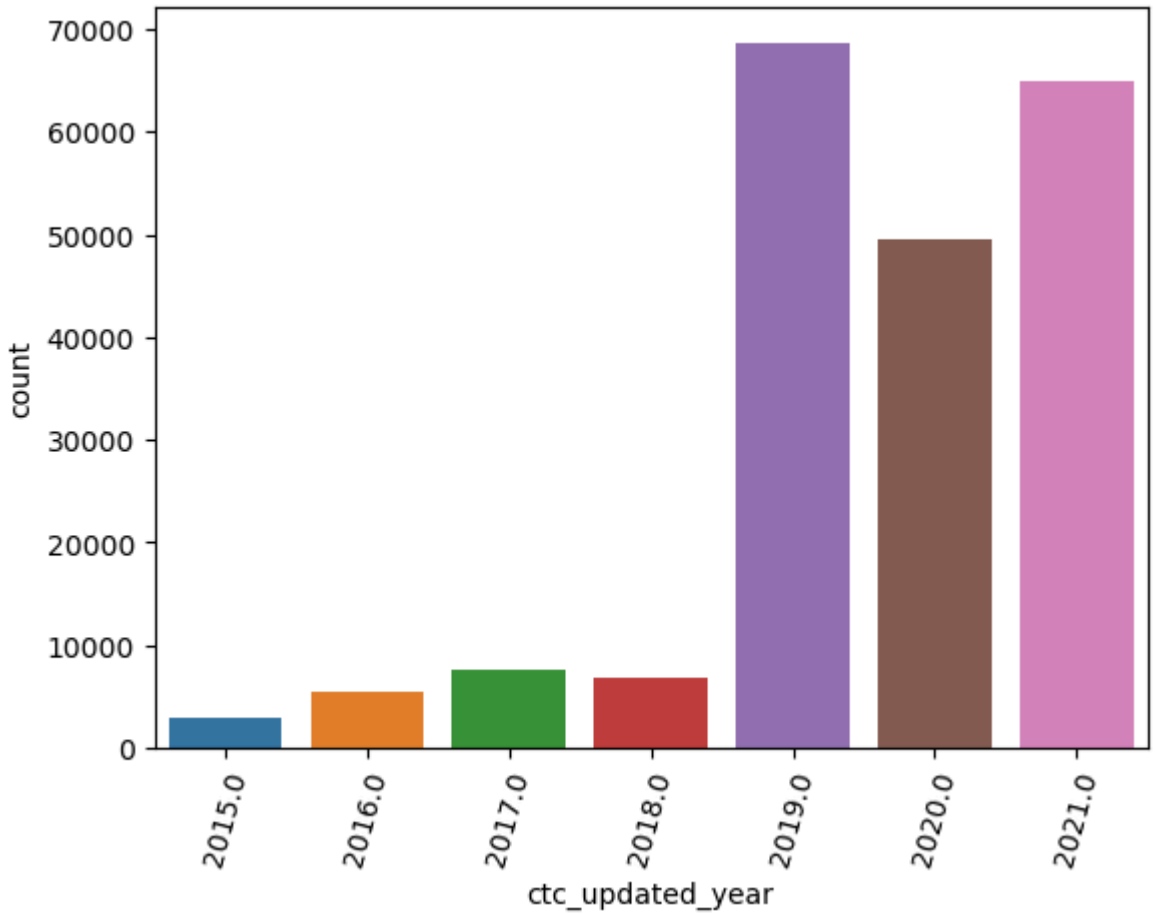
Dropping the Outlier from the CTC column

In [21]:
```
data = data[data['ctc'] <= 1000000000]
```

In [22]:
```
plt.figure(figsize=(15, 8))
sns.countplot(data = data, x = 'orgyear')
plt.xticks(rotation = 75)
plt.show()
```

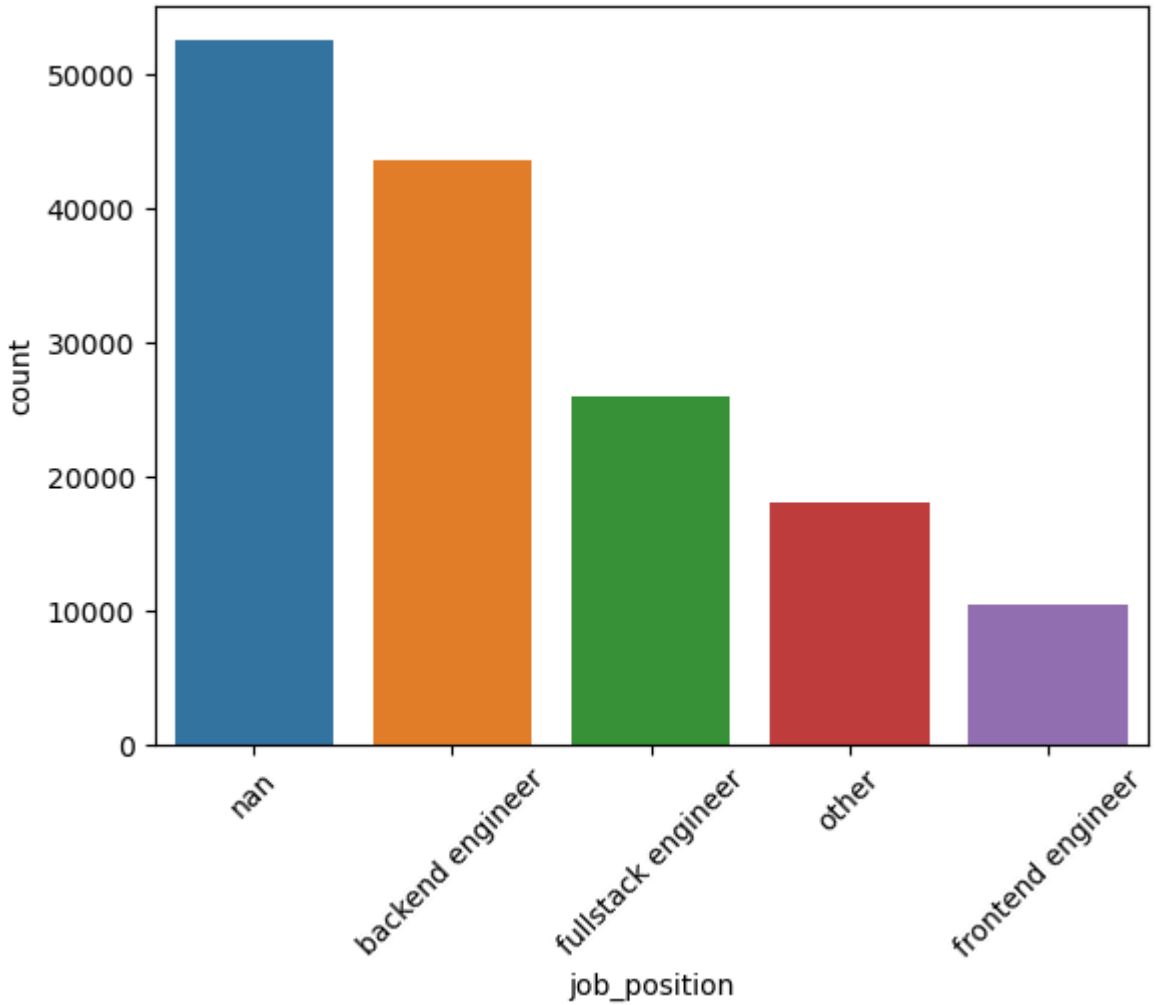


In [23]:
```
sns.countplot(data = data, x = 'ctc_updated_year')
plt.xticks(rotation = 75)
plt.show()
```



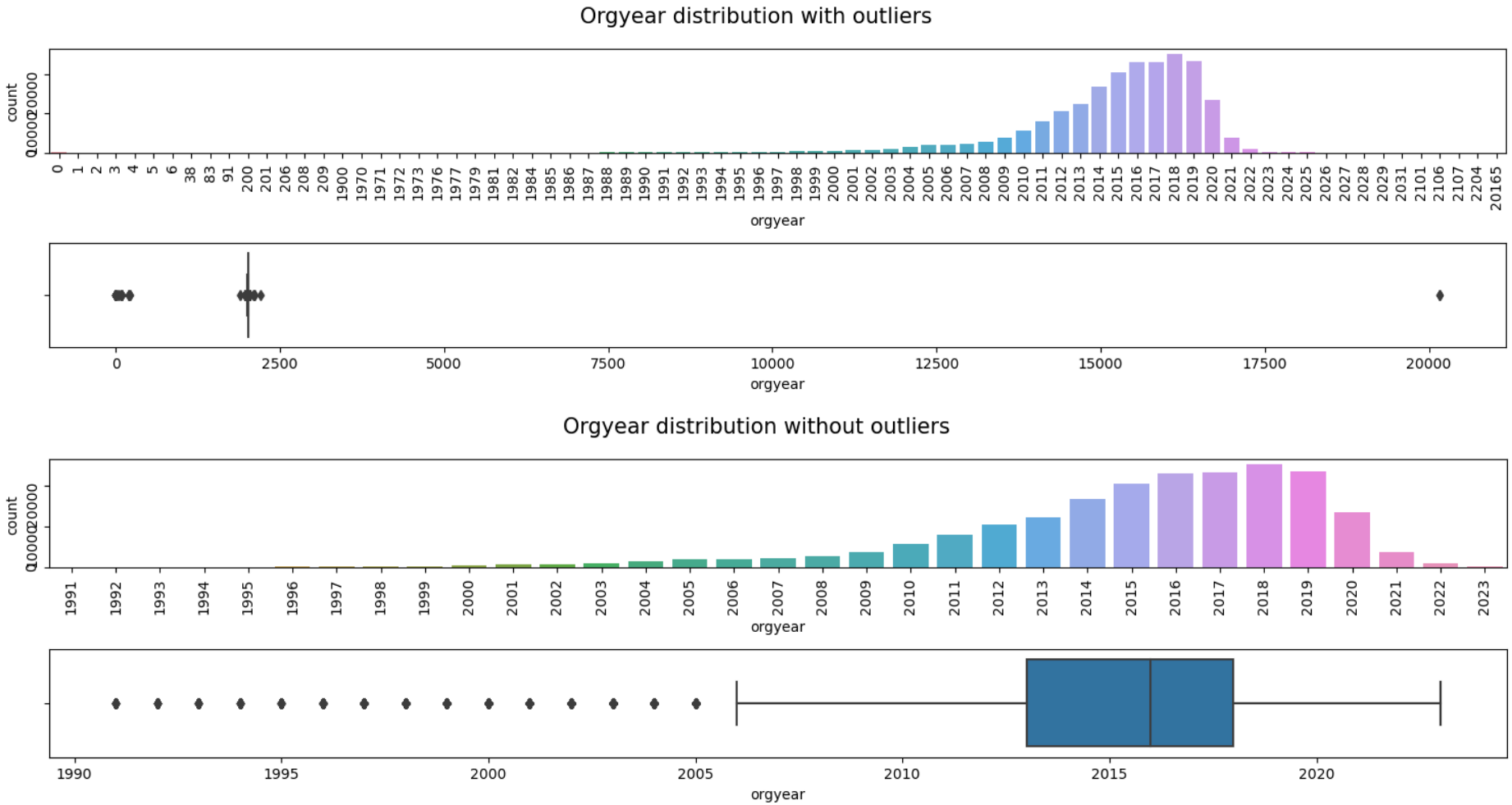
In [24]:
```
data1 = data['job_position'].value_counts(ascending = False).head(5).reset_index()
```

In [25]:
```
sns.barplot(data = data1, x = 'job_position', y = 'count')
plt.xticks(rotation = 45)
plt.show()
```



In [26]:
```
df = data['orgyear']
fig, axs = plt.subplots(2,1,figsize=(15,4))
sns.countplot(ax = axs[0], x=df)
axs[0].tick_params(labelrotation=90)
sns.boxplot(ax = axs[1], x=df)
fig.suptitle('Orgyear distribution with outliers', fontsize=15)
plt.tight_layout()
plt.show()
lower_bound = data['orgyear'].quantile(0.001)
upper_bound = data['orgyear'].quantile(0.999)
df = df[(df >= lower_bound) & (df <= upper_bound)]
```

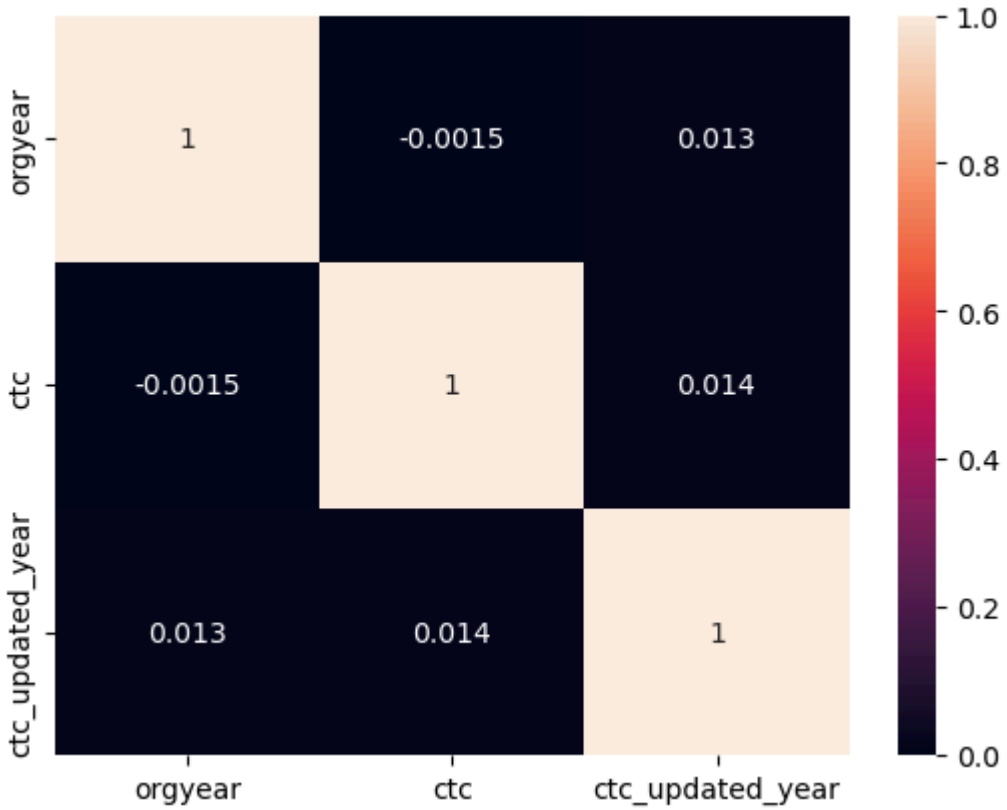
```
fig, axs = plt.subplots(2,1,figsize=(15,4))
sns.countplot(ax = axs[0], x=df)
axs[0].tick_params(labelrotation=90)
sns.boxplot(ax = axs[1], x=df)
fig.suptitle('Orgyear distribution without outliers', fontsize=15)
plt.tight_layout()
plt.show()
```



Bivariate Analysis

In [27]:
```
corr = data.corr(numeric_only=True)
sns.heatmap(data = corr, annot = True)
```

Out[27]:
```
<Axes: >
```



Feature Engineering

In [28]:
```
current_year = datetime.now().year
data['YOE'] = current_year - data['orgyear']
```

In [29]:
```
data.shape
```

Out[29]:
```
(205744, 7)
```

In [30]:
```
data.duplicated().value_counts()
```

Out[30]:
```
False    205601
True        143
Name: count, dtype: int64
```

In [31]:
```
data.drop_duplicates(inplace=True)
data.shape
```

Out[31]:
```
(205601, 7)
```

- Dropped the null values from Company hash and job position column
- Majority of the people had joined from 2015 to 2019
- Backend Engineer and Full stack engineers are more in company as compaed to other roles
- Most of the people increment was done in 2019

Renaming compnay's name to other which are having count less than 5

In [32]:
```
company_counts = data['company_hash'].value_counts()
mask = data['company_hash'].map(company_counts) > 5
data.iloc[~mask.values, data.columns.get_loc('company_hash')] = 'Others'
```

In [33]:
```
grouped_CTC  = data.groupby(['YOE','job_position','company_hash'])['ctc'].describe()
data_merge=data.merge(grouped_CTC, on=['YOE','job_position','company_hash'], how='left')
data_merge.head()
```

Out[33]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	YOE	count	mean	
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016	1100000	other	2020.0	9	1.0	1.100000e+06	
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018	449999	fullstack engineer	2019.0	7	7.0	7.742856e+05	2.50
2	Others	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	2015	2000000	backend engineer	2020.0	10	923.0	1.444876e+06	4.87
3	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	2017	700000	backend engineer	2019.0	8	7.0	1.158571e+06	4.04
4	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	2017	1400000	fullstack engineer	2019.0	8	1.0	1.400000e+06	

In [34]:
```
def segment(a,b_50,b_75):
    if a<b_50:
        return 3
    elif a>=b_50 and a<=b_75:
        return 2

    elif a>=b_75:
        return 1
data_merge['designation'] =data_merge.apply(lambda x: segment(x['ctc'],x['50%'],x['75%']),axis=1)
```

In [35]:
```
df = data_merge[(data_merge['ctc'] > data_merge['75%']) & (data_merge['designation']) == 1]
```

Question

1. Top 10 employees (earning more than most of the employees in the company)

In [36]:
```
top_10_data = df.nlargest(10, 'ctc')
top_10_data
```

Out[36]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	YOE	count	m
117546	obvqnuqxdwgb	5b4bed51797140db4ed52018a979db1e34cee49e27b488...	2018	255555555	nan	2016.0	7	2.0	1.289778e
3299	Others	06d231f167701592a69cdd7d5c825a0f5b30f0347a4078...	2021	250000000	nan	2020.0	4	209.0	7.852931e
106	Others	996aef9bba62bd99d6cb8e8c112c0ec8096b203ae50b97...	2017	200000000	support engineer	2020.0	8	93.0	5.882441e
301	Others	2b649949f0a00c444db6ae38d66a972e37cc3a90ba65a0...	2017	200000000	qa engineer	2020.0	8	197.0	2.217289e
361	Others	e1dfef2de8d773780471295f756a0c2957cbe368e33216...	2014	200000000	engineering leadership	2019.0	11	122.0	3.800082e
487	Others	eda5097c113cc6e8ff663ca9764c78e35d3eec75bdcea9...	2014	200000000	qa engineer	2020.0	11	279.0	1.502885e
602	xzegojo	4368cc6185184b811c3a4b9cef05dd1e45a682a6e94056...	2017	200000000	nan	2020.0	8	216.0	1.478828e
691	Others	dfdb45fb9631b9064a94be87a27a621068530ac1f3807c...	2017	200000000	other	2020.0	8	643.0	6.290416e
720	Others	261f76b9954ebe5e6dc102b0cd5847354cf27112f8a422...	2015	200000000	frontend engineer	2020.0	10	358.0	2.303805e
734	vwwtznhqt	0f7322f8f4423e695df58edb4f002dac637d8de021373a...	2013	200000000	other	2020.0	12	23.0	9.793043e

- Top 10 employees of data science in each company earning more than their peers

In [37]:
```
filtered_employees = data_merge[(data_merge['job_position'] == 'data scientist') & (data_merge['designation'] == 1)]
top_10_data_science_per_company = filtered_employees.groupby('company_hash', group_keys=False).apply(lambda x: x.nlargest(10, 'ct
top_10_data_science_per_company
```

Out[37]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	YOE	count	me
2687	Others	72ed7ced98573f71c8f95bc8b75aac4f0677e8872c6bec...	2019	199800000	data scientist	2020.0	6	140.0	2.398252e+
1736	Others	ee8dd42d6ea8365909147d861c7978d19f727a8075ba96...	2020	102500000	data scientist	2020.0	5	51.0	5.094412e+
10311	Others	2e1d492bc09bfe0d4cc9757a9c63a296c1527af1c8ecc8...	2021	100000000	data scientist	2020.0	4	25.0	5.234720e+
13507	Others	e7722fb701c61e5cad82c39ee8bf3debe160d429b72c64...	2015	100000000	data scientist	2020.0	10	205.0	1.780620e+
32197	Others	5dd32aca5f483b8fa4d539778bb3f0a24073a93a80dd5a...	2010	100000000	data scientist	2020.0	15	56.0	3.597321e+
...	...	...	...	...	...	...	...	...	
172958	zvz	f2be350614a51edd33a53f201374ae734457509d3f151b...	2016	1600000	data scientist	2019.0	9	4.0	1.142500e+
52678	zxtrotz	751b1fb94f9054ecc14b44ebf91c3cbd92a47ea0194492...	2013	3000000	data scientist	2019.0	12	2.0	2.045000e+
99020	zxtrotz	515862ad8c8c33263846231044741bfc177af2cddcf00f...	2018	1000000	data scientist	2021.0	7	2.0	6.650000e+
28673	zxtrotz	2182cd4f16b2a915d6c53f901f9911bb6b3b22caaaa0ae...	2017	750000	data scientist	2019.0	8	2.0	7.000000e+
111745	zxtrotz	163e546c8418ddc4b300471c1472044f582cd3be008da1...	2015	700000	data scientist	2019.0	10	3.0	6.666667e+

548 rows × 16 columns

- Bottom 10 employees of data science in each company earning less than their peers - Class 3

In [38]:
```
filtered_employees = data_merge[(data_merge['job_position'] == 'data scientist') & (data_merge['designation'] == 3)]
bottom_10_data_science_per_company = filtered_employees.groupby('company_hash', group_keys=False).apply(lambda x: x.nsmallest(10,
bottom_10_data_science_per_company
```

Out[38]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	YOE	count	mean
168157	Others	05801a432a038c254972e356598ca6aa139a18c31d6611...	2021	4000	data scientist	2020.0	4	25.0	5.234720e+06
193975	Others	585f7e9865dcdcaad7edf10909d796ba2c5210cde3530b...	2017	4000	data scientist	2018.0	8	266.0	9.182131e+05
136846	Others	e374eea75640881206a21894f69190138c2c0535277dc1...	2017	7000	data scientist	2019.0	8	266.0	9.182131e+05
24091	Others	ab2dc9db23c3104f0b6b3dbd4cdd5bfb9e5829b8b7943d...	2017	7200	data scientist	2019.0	8	266.0	9.182131e+05
183038	Others	287dd26e9357888e0ba2c7482764131f7bbcb1748a4f56...	2019	7250	data scientist	2020.0	6	140.0	2.398252e+06
...	...	...	...	...	...	...	...	...	..
202816	zvz	4d81b7fea5707674ecd08571144b503e625ea1311be381...	2016	600000	data scientist	2021.0	9	4.0	1.142500e+06
187363	zvz	af7af44e1788f7134691bc3782a8256fb07ef9ac3a51e1...	2019	620000	data scientist	2020.0	6	9.0	1.385556e+06
9340	zxtrotz	8db7199e084be127053249086830cf3cda7f595e883a22...	2018	330000	data scientist	2020.0	7	2.0	6.650000e+05
98324	zxtrotz	d5d7fa93cf62d046654e21716c7bdd613e5f559b47bc21...	2017	650000	data scientist	2019.0	8	2.0	7.000000e+05
56100	zxtrotz	2b7979bb62110fbb5e97f3f7f25d79f102536d3b84d538...	2013	1090000	data scientist	2019.0	12	2.0	2.045000e+06

590 rows × 16 columns

1. Bottom 10 employees (earning less than most of the employees in the company)- Tier 3

In [39]:
```
data_merge[(data_merge['ctc'] > data_merge['75%']) & (data_merge['designation']) == 3]
bottom_10_data = data_merge.nsmallest(10, 'ctc')
bottom_10_data
```

Out[39]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	YOE	count	mean	
135315	xzntqcxtfmxn	3505b02549ebe2c95840ac6f0a35561a3b4cbe4b79cdb1...	2014	2	backend engineer	2019.0	11	2.0	1.000001e+06	1
118146	xzntqcxtfmxn	f2b58aeed3c074652de2cfd3c0717a5d21d6fbcf342a78...	2013	6	nan	2018.0	12	2.0	1.000000e+01	5
114082	xzntqcxtfmxn	23ad96d6b6f1ecf554a52f6e9b61677c7d73d8a409a143...	2013	14	nan	2018.0	12	2.0	1.000000e+01	5
184750	Others	b8a0bb340583936b5a7923947e9aec21add5ebc50cd60b...	2016	15	nan	2018.0	9	1141.0	2.197371e+06	1
183608	Others	75357254a31f133e2d3870057922feddeba82b88056a07...	2019	16	nan	2018.0	6	1528.0	1.760520e+06	1
54788	Others	8786759b95d673466e94f62f1b15e4f8c6bd7de6164074...	2020	24	other	2020.0	5	223.0	4.259803e+06	1
91495	Others	512f761579fb116e215cabc9821c7f81153f0763e16018...	2016	25	android engineer	2018.0	9	300.0	8.127241e+05	1
116859	Others	f7e5e788676100d7c4146740ada9e2f8974defc01f571d...	2022	200	nan	2021.0	3	62.0	4.271326e+06	1
166228	Others	c411a6917058b50f44d7c62751be9b232155b23211de4c...	2013	300	database administrator	2019.0	12	11.0	9.041845e+06	2
81981	Others	edcfb902656b736e1f35863298706d9d34ee795b7ed85a...	2018	500	cofounder	2019.0	7	32.0	7.632969e+05	6

1. Top 10 employees in each company - X department - having 5/6/7 years of experience earning more than their peers - Tier X

In [40]:
```
df = data_merge[(data_merge['YOE'] > 5)]
top_10_data = df.nlargest(10, 'ctc')
top_10_data
```

Out[40]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	YOE	count	m
117546	obvqnuqxdwgb	5b4bed51797140db4ed52018a979db1e34cee49e27b488...	2018	255555555	nan	2016.0	7	2.0	1.289778e
106	Others	996aef9bba62bd99d6cb8e8c112c0ec8096b203ae50b97...	2017	200000000	support engineer	2020.0	8	93.0	5.882441e
301	Others	2b649949f0a00c444db6ae38d66a972e37cc3a90ba65a0...	2017	200000000	qa engineer	2020.0	8	197.0	2.217289e
361	Others	e1dfef2de8d773780471295f756a0c2957cbe368e33216...	2014	200000000	engineering leadership	2019.0	11	122.0	3.800082e
487	Others	eda5097c113cc6e8ff663ca9764c78e35d3eec75bdcea9...	2014	200000000	qa engineer	2020.0	11	279.0	1.502885e
602	xzegojo	4368cc6185184b811c3a4b9cef05dd1e45a682a6e94056...	2017	200000000	nan	2020.0	8	216.0	1.478828e
691	Others	dfdb45fb9631b9064a94be87a27a621068530ac1f3807c...	2017	200000000	other	2020.0	8	643.0	6.290416e
720	Others	261f76b9954ebe5e6dc102b0cd5847354cf27112f8a422...	2015	200000000	frontend engineer	2020.0	10	358.0	2.303805e
734	vwwtznhqt	0f7322f8f4423e695df58edb4f002dac637d8de021373a...	2013	200000000	other	2020.0	12	23.0	9.793043e
836	mqxonrtwgzt v bvyxzaqv sqghu wgbuvzj	cda8d723438e81185d2ee8c348870a4612eea974cdb2db...	2017	200000000	data scientist	2020.0	8	1.0	2.000000e

1. Top 10 companies (based on their CTC)

In [41]:
```
company_ctc_totals = data_merge.groupby('company_hash')['ctc'].sum()
top_10_companies = company_ctc_totals.sort_values(ascending=False).head(11)
top_10_companies
```

Out[41]:
```
company_hash
Others                        128739371403
nvnv wgzohrnvzwj otqcxwto      13765174109
vbvkgz                         11798242055
zgn vuurxwvmrt vwwghzn          8121636481
xzegojo                         7579607874
bxwqgogen                       7006849399
vwwtznhqt                       6897730775
fxuqg rxbxnta                   6062971147
zgn vuurxwvmrt                  5941526113
wgszxkvzn                       5635685321
gqvwrt                          4164552184
Name: ctc, dtype: int64
```

1. Top 2 positions in every company (based on their CTC)

In [42]:
```
position_ctc_totals = data_merge.groupby(['company_hash', 'job_position'])['ctc'].sum().reset_index()
top_2_positions_per_company = position_ctc_totals.groupby('company_hash', group_keys=False).apply(lambda x: x.nlargest(2, 'ctc'))
top_2_positions_per_company
```

Out[42]:

	company_hash	job_position	ctc
149	Others	other	28152567494
145	Others	nan	21211074933
294	a ntwyzgrgsxto	nan	7375000
295	a ntwyzgrgsxto	other	7150000
300	aaqxctz avnv owxtzwto vzvrjnxwo ucn rna	other	3600000
...	...	...	...
23855	zxyxrtzn ntwyzgrgsxto	other	1660000
23858	zxzlvwvqn	backend engineer	29560000
23863	zxzlvwvqn	nan	20820000
23868	zxztrtvuo	backend engineer	26830000
23876	zxztrtvuo	nan	20415000

6307 rows × 3 columns

In [43]:
```
data_merge.drop(columns = ['email_hash','count','mean','std','min','25%','50%','75%','max'], inplace = True)
```

Using Target Encoding on Job Position and Company hash column

In [44]:
```
job_position_target_mean = data_merge.groupby('job_position')['ctc'].mean()
data_merge['job_position'] = data_merge['job_position'].map(job_position_target_mean)

company_hash_target_mean = data_merge.groupby('company_hash')['ctc'].mean()
data_merge['company_hash'] = data_merge['company_hash'].map(company_hash_target_mean)
```

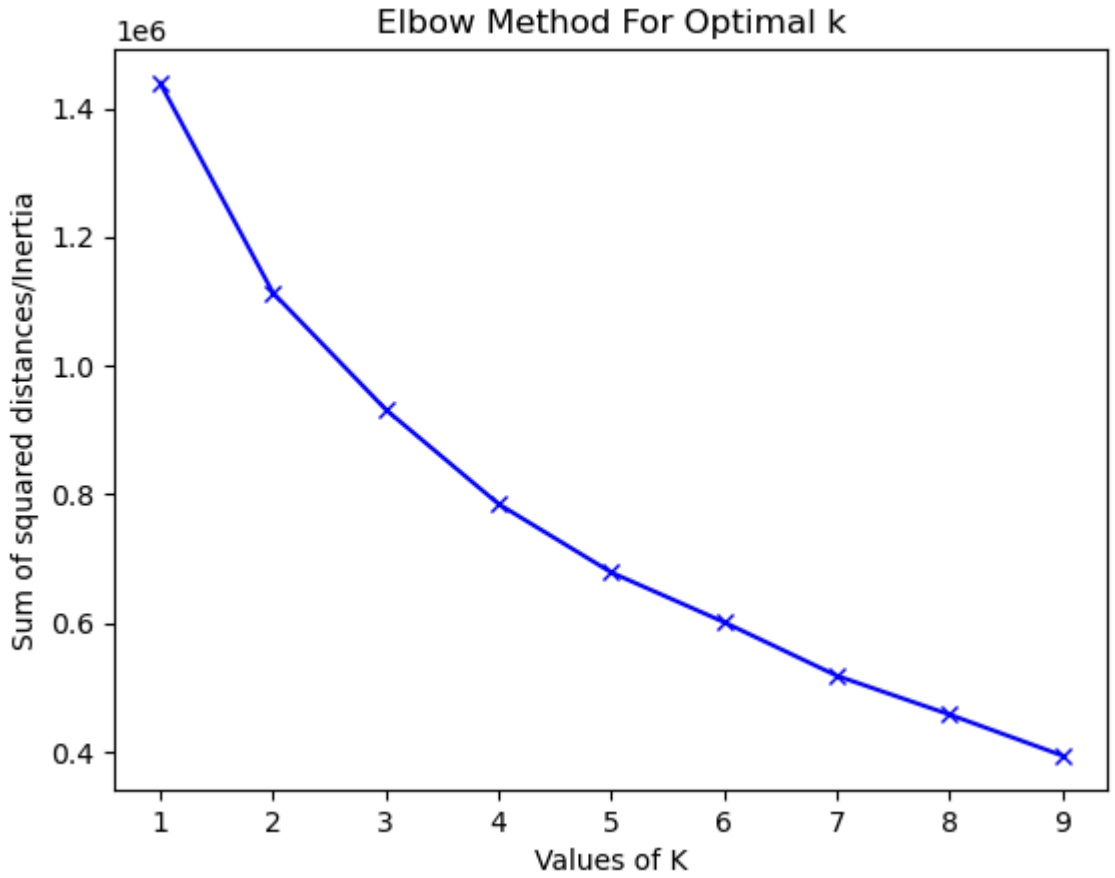
Standardizing the Data

In [45]:
```
df = data_merge.copy()
```

In [46]:
```
scaler = StandardScaler()
data_merge = scaler.fit_transform(data_merge)
```

K- Menas

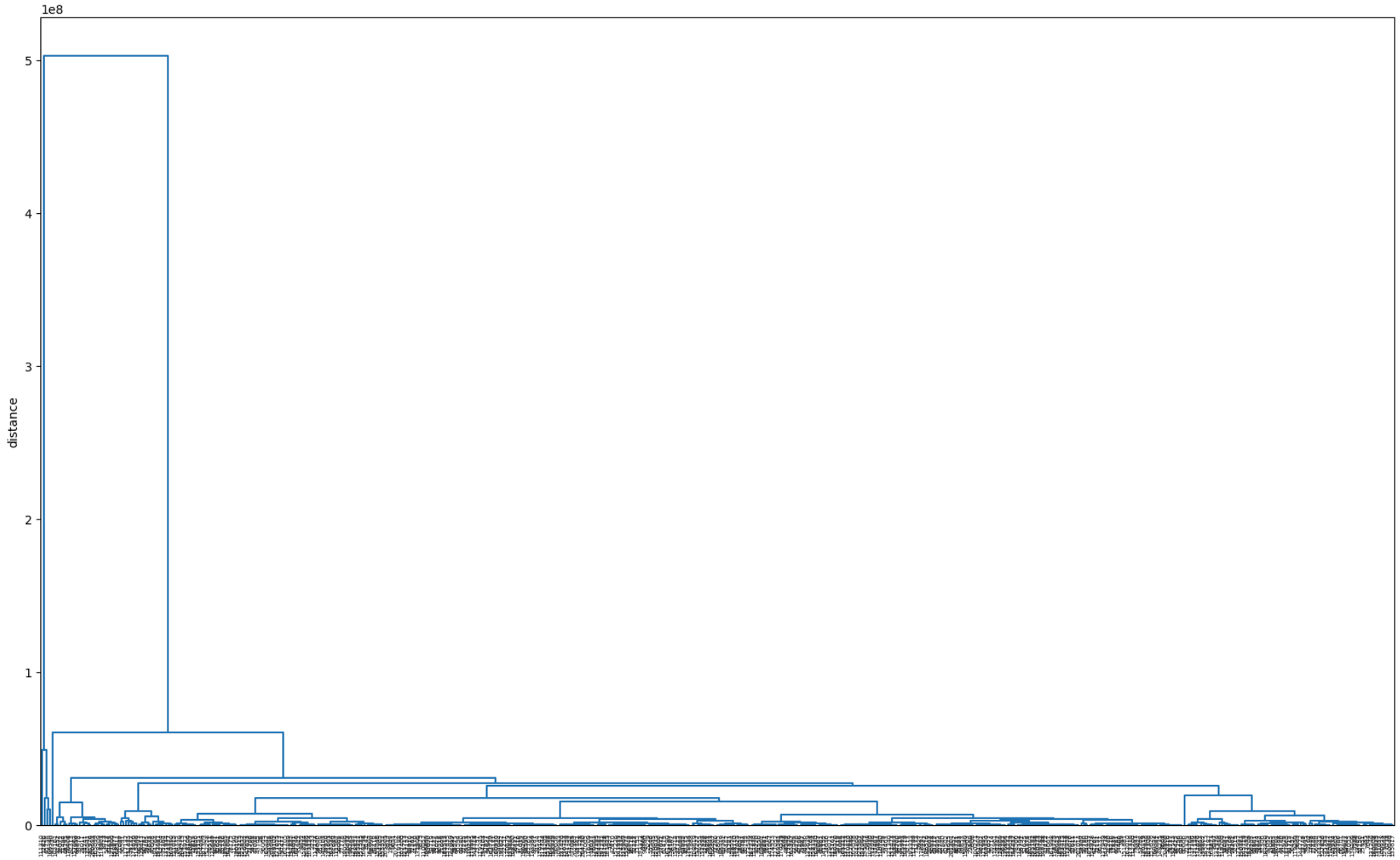
In [47]:
```
Sum_of_squared_distances = []
K = range(1,10)
for num_clusters in K :
  kmeans = KMeans(n_clusters=num_clusters)
  kmeans.fit(data_merge)
  Sum_of_squared_distances.append(kmeans.inertia_)
plt.plot(K,Sum_of_squared_distances,'bx-')
plt.xlabel('Values of K')
plt.ylabel('Sum of squared distances/Inertia')
plt.title('Elbow Method For Optimal k')
plt.show()
```



Hirarical Clustring

In [48]:
```
sample = df.sample(500)
Z = hc.linkage(sample, method='ward')

fig, ax = plt.subplots(figsize=(20, 12))
hc.dendrogram(Z, labels=sample.index, ax=ax, color_threshold=2)
plt.xticks(rotation=90)
ax.set_ylabel('distance')
plt.show()
```



- Business Implications:
  - Growth and Talent Retention:

    -- The hiring surge from 2015 to 2019 aligns with the dominance of technical roles. This highlights the company's reliance on skilled engineers, especially Backend and Full-Stack Engineers.

-- Targeted retention strategies for these key roles are essential, given their importance to the company’s success.

- Performance-Driven Culture:

  -- The increments in 2019 could emphasize the company's focus on rewarding performance. Maintaining consistent reviews and adjusting salaries as per market standards can help sustain employee satisfaction.

- Future Planning:

  -- The hiring peak in 2015–2019 may lead to a plateau or declining trend in new hires as the company stabilizes its workforce.